

CPSC 457 – Principles of Operating Systems
Daniel de Castro
Tutorial 1: Apache Subversion (Estimate time: 10-15 minutes)
January 17, 2012

This practice was developed to demonstrate some basic SVN commands. It also shows how to create public and private keys used for authentication. For the purpose of this exercise, these keys will be our “project”. We will create a pair of keys, create our SVN repository and then make some usual operations over our repository.

1. Create and access the directory for our project

```
$ mkdir -p ~/cpsc457/tut01  
$ cd ~/cpsc457/tut01
```

NOTES:

- In Linux, the symbol ~ (tilde) represents the user's home directory. Equivalent to \$HOME.
- Q: What is that option “-p” in mkdir? A: “man mkdir” ;)

2. Create your keys

```
$ cd ~/cpsc457/tut01  
$ ssh-keygen -t rsa -b 1024 -f YOURNAME.key
```

This command will create two files:

- YOURNAME.key is your private key.
- YOURNAME.key.pub is your public key.

And, again, you should also try “man ssh-keygen” for the options.

3. Create a repository

To create a repository, type the following command on a terminal window.

```
$ svnadmin create ~/svn
```

This command creates the directory “svn” in your home directory and some files used by SVN.

NOTE: For this course, the instructor will create the students' repositories.

4. Create the directory structure for your project

```
$ svn mkdir file:///~/svn/trunk
```

This command will open a program for you to describe what is the modification you are making to the project. To save your description press CTRL-X.

To avoid this, use the option “-m” as shown below.

CPSC 457 – Principles of Operating Systems
Daniel de Castro
Tutorial 1: Apache Subversion (Estimate time: 10-15 minutes)
January 17, 2012

Spoiler Alert: You will receive an error. SVN sends “~” to the “server”, that cannot process that. It will also generate a file (usually “svn-commit.tmp”) that you should remove. Instead of “~”, you should use \$HOME (or type the complete path), as follows:

```
$ svn mkdir file://$HOME/svn/trunk -m "Creating directory trunk"
$ svn mkdir file://$HOME/svn/tags -m "Creating directory tags"
$ svn mkdir file://$HOME/svn/branches "Creating directory branches"
```

5. Upload your project to the repository

```
$ cd ~/cpssc457
$ svn import tut01 file://$HOME/svn/trunk/tut01 -m "Initial import"
```

Import will upload your files to the SVN server. If you cancel the command during the process, no file will be available in the server.

Notice that we are tut01 as a “subproject” in our repository, i.e., we won't create all the directories (trunk, tags, branches) for each tutorial.

6. List repository contents

```
$ svn ls file://$HOME/svn
$ svn ls -R file://$HOME/svn
```

7. Create the working directory

```
$ mkdir -p ~/cpssc457/work
$ cd ~/cpssc457/work
$ svn checkout file://$HOME/svn/trunk/tut01
```

NOTE: This directory “trunk” you will use to make modifications in your project. You could even remove the original directory (~/cpssc457/tut01).

8. Get information about your project

```
$ cd ~/cpssc457/work/tut01
$ svn info
$ svn status
```

Use “info” for information about the repository or the working directory and “status” to get each file's status (comparing the working directory to the repository). Status won't return nothing, as there is no difference from the repository.

CPSC 457 – Principles of Operating Systems
Daniel de Castro
Tutorial 1: Apache Subversion (Estimate time: 10-15 minutes)
January 17, 2012

9. Edit your project

Create a file within your working directory (e.g, ~/cpsc457/work/tut01/README).

10. Check the files' status, again (you should be at ~/cpsc456/work/tut01)

“?” indicates an unkown file

11. Add the file

```
$ cd ~/cpsc457/work/tut01
$ svn add README
```

12. Check status once more and verify what changed

13. Save changes to the repository

```
$ cd ~/cpsc457/work/tut01
$ svn commit -m "Saving my readme"
```

14. Check status one last time

15. Check out a previous version

```
$ mkdir ~/cpsc457/old
$ cd ~/cpsc457/old
$ svn checkout -r 4 file://$HOME/svn/trunk
```

Now, compare the directory old with your working directory and look for differences.

16. Update your project

```
$ cd ~/cpsc457/work/tut01
$ svn update
```

Updates are important when working in teams. If someone else performed any changes to the repository, you need to bring those changes to your working directory, to keep consistency. In this practice, however, you are working by yourself, so the update will not cause changes to your working directory.

17. Get help

```
$ svn help
$ svn help help | less
```